AMENDMENTS TO THE CLAIMS

Please cancel claim 25.

Please amend claims 16, 17, 19, 28, 33, 40, 45 and 46 as follows.

1-15. (canceled)

16. (currently amended) A method of mapping graphical block diagram block parameters in a graphical block diagram modeling environment, comprising:

receiving a user-defined block parameter representing a value to be used during block diagram execution; and

processing the user-defined block parameter to ~~optimally~~ produce a run-time block parameter, wherein the run-time block parameter reduces memory requirements for executing a block diagram.

17. (currently amended) The method of claim 16, further comprising a block method inversely mapping the block run-time parameter to the user-defined block parameter ~~to optimize block implementation~~.

18. (previously presented) The method of claim 16, further comprising receiving a plurality of user-defined block parameters.

19. (currently amended) The method of claim 18, further comprising processing the plurality of user-defined block parameters to ~~optimally~~ produce a run-time block parameter.

20. (previously presented) The method of claim 19, wherein the plurality of user-defined block parameters is processed to produce a single run-time block parameter.

21. (previously presented) The method of claim 19, wherein the run-time block parameter is configured to return at least one of simulation results, and automatically generated code that implements graphical block diagram model equations.

22. (previously presented) The method of claim 19, further comprising mapping by discarding at least a portion of the plurality of user-defined block parameters to reduce memory requirements.

23. (previously presented) The method of claim 19, further comprising pooling like non-interfaced run-time block parameters to reduce repetition of the non-interfaced run-time block parameters.

24. (previously presented) The method of claim 23, wherein the pooling step comprises mapping user-defined block parameters into an existing pool.

25. (canceled)

26. (previously presented) The method of claim 19, further comprising mapping by translating the plurality of user-defined block parameters based at least in part on type.

27. (previously presented) The method of claim 16, wherein the run-time block parameter is configured to return at least one of simulation results, and automatically generated code that implements graphical block diagram model equations.

28. (currently amended) The method of claim 27, wherein when the code is automatically generated, ~~the~~ parameter expressions are maintained in the automatically generated code.

29. (previously presented) The method of claim 28, wherein the parameter expressions contain interfaced variables that are updatable during modeling.

30. (previously presented) The method of claim 29, further comprising converting to a relatively more compact representation portions of the parameter expressions that are constants while providing access to interfaced variables that are updatable.

31. (previously presented) The method of claim 29, wherein interfaced variables are updatable.

32. (previously presented) The method of claim 31, wherein updatable variables used in a plurality of blocks are declared only once.

33. (currently amended) A method of mapping graphical block diagram block parameters in a graphical block diagram modeling environment, comprising:

   receiving a plurality of user-defined block parameters;

   processing the plurality of user-defined block ~~parameter~~ parameters to ~~optimally~~ produce a plurality of run-time block parameters;

   pooling together like non-interfaced run-time block parameters to ~~create a run-time parameter expression for use during modeling~~ reuse data for the like non-interfaced run-time block parameters.

34. (previously presented) The method of claim 33, wherein pooling further comprises mapping user-defined block parameters into an existing pool.

35. (previously presented) The method of claim 33, wherein the non-interfaced run-time block parameters have stored values that differ from presented values.

36. (previously presented) The method of claim 35, wherein the non-interfaced run-time block parameters are fixed point.

37. (previously presented) The method of claim 33, further comprising translating at run-time constant parameter values to an internal representation to enable increased pooling.

38. (previously presented) The method of claim 33, wherein the step of pooling further comprises collecting constant portions of an expression containing an interfaced variable.

39. (previously presented) The method of claim 33, wherein the run-time block parameters are configured to return at least one of simulation results, and automatically generated code that implements graphical block diagram model equations.

40. (currently amended) The method of claim 39, wherein when the code is automatically generated, ~~the~~ parameter expressions are maintained in the automatically generated code.

41. (previously presented) The method of claim 40, wherein the parameter expressions contain interfaced variables which are updatable.

42. (previously presented) The method of claim 41, further comprising converting to a relatively more compact representation portions of the parameter expressions that are constants while providing access to interfaced variables.

43. (previously presented) The method of claim 41, wherein interfaced variables are updatable.

44. (previously presented) The method of claim 43, wherein updatable variables used in a plurality of blocks are declared only once.

45. (currently amended) A medium for use in a graphical modeling environment on an electronic device, the medium holding instructions executable using the electronic device for performing a method of mapping graphical block diagram block parameters, the method comprising:

    receiving a user-defined block parameter representing a value to be used during block diagram execution; and

    processing the user-defined block parameter to ~~optimally~~ produce a run-time block parameter for use during ~~modeling~~ execution, wherein the run-time block parameter reduces memory requirements for executing a block diagram.

46. (currently amended) A medium for use in a graphical modeling environment on an electronic device, the medium holding instructions executable using the electronic device for performing a method of mapping graphical block diagram block parameters, the method comprising:

    receiving a plurality of user-defined block parameters;

    processing the plurality of user-defined block ~~parameter~~ parameters to ~~optimally~~ produce a plurality of run-time block parameters; and

pooling together like non-interfaced run-time block parameters to ~~create a run-time parameter expression for use during modeling~~ <u>reuse data for the like non-interfaced run-time block parameters</u>.